

《数据库系统》课程设计

--高考志愿辅助填报系统设计与开发

|  |  |
| --- | --- |
| 班级： | 软件2202 |
| 学号： | 2231052217 |
| 姓名： | 张元嘉 |

2024年 XX月XX日

数据库系统课程设计评阅

|  |  |  |
| --- | --- | --- |
| 一、系统分析与设计（60分） | | 实际得分 |
|  | |  |
| 1.1需求分析：给出完整的数据需求、系统功能图,业务流程图,数据流程图,和数据字典等，分析详细、正确。 | 满 分  （20分） |  |
| 1.2概念结构设计：有详细的分析步骤和理论依据，E-R图正确、完备。 | 满 分  （15分） |  |
| 1.3逻辑结构设计: 有详细的转换步骤和理论依据，PD中概念模型图正确，并确定主码、外码。 | 满 分  （15分） |  |
| 1.4物理结构设计:PD中物理模型图正确，有索引、聚簇、日志、备份等的存储安排和存储结构，内容正确。 | 满 分  （10分） |  |
| 二、系统实现及界面（30分） | |  |
| 2.1系统运行正常，代码正确，根据数据安全性和完整性需要，完成增、删、改查、汇总等功能。 | 满 分  （15分） |  |
| 2.2系统界面美观，布局合理，根据系统需求，实现数据安全性和完整性，实现对象完备、正确。 | 满 分  （15分） |  |
| 三、系统评价（10分） | | |
| 3.1评价、总结与体会；  3.2分析、展望。已解决和尚未解决的问题分析，完善的设想与建议。 | 满 分  （10分） |  |
| 总分 | |  |
| 阅卷教师签名 | |  |

目录

[一、 系统需求分析 1](#_Toc167484685)

[（一） 需求概述 1](#_Toc167484686)

[1. 用户管理 1](#_Toc167484687)

[2. 学生功能 1](#_Toc167484688)

[（二） 业务流分析 1](#_Toc167484689)

[（三） 数据流分析 3](#_Toc167484690)

[（四） 数据字典 4](#_Toc167484691)

[1. 数据项 4](#_Toc167484692)

[2. 数据结构 5](#_Toc167484693)

[3. 数据流 6](#_Toc167484694)

[二、 数据库概念结构设计 6](#_Toc167484695)

[（一） 实体分析 6](#_Toc167484696)

[（二） 属性分析 6](#_Toc167484697)

[（三） 联系分析 7](#_Toc167484698)

[（四） 概念模型分析（.CDM图） 7](#_Toc167484699)

[三、 数据库逻辑结构设计 8](#_Toc167484700)

[(一) 概念模型转化为逻辑模型 8](#_Toc167484701)

[1. 一对一关系的转化 8](#_Toc167484702)

[2. 一对多关系的转化 8](#_Toc167484703)

[3. 多对多关系的转化 8](#_Toc167484704)

[(二) 逻辑模型设计（.LDM图） 8](#_Toc167484705)

[四、 数据库物理结构设计 9](#_Toc167484706)

[(一) 表设计 9](#_Toc167484707)

[(二) 创建表和完整性约束代码设计 11](#_Toc167484708)

[(三) 创建物理模型设计图（.PDM） 13](#_Toc167484709)

[(四) 视图、索引、存储过程和触发器 13](#_Toc167484710)

[五、 数据库功能实现及界面展示 18](#_Toc167484711)

[(一) 登录模块 18](#_Toc167484712)

[(二) 学生模块 19](#_Toc167484713)

[(三) 管理员模块 22](#_Toc167484714)

[六、 总结与展望 24](#_Toc167484715)

[总结 24](#_Toc167484716)

[展望 24](#_Toc167484717)

1. 系统需求分析
2. 需求概述

1. 用户管理

注册： 学生和管理员可以注册账号。登录： 学生和管理员可以通过用户名和密码登录系统。

2. 学生功能填报志愿： 学生可以根据自己的成绩和兴趣选择志愿，系统应提供志愿填报页面。查看成绩： 学生可以查看自己的高考成绩。修改志愿： 学生可以在修改自己的志愿3. 管理员功能录入信息： 管理员可以录入学校专业信息。查看志愿情况： 管理员可以查看学生填报的志愿情况，包括各个学校、专业的报考信息。修改学校信息：管理员可以修改学校专业信息

4. 系统管理学生信息管理： 管理员可以对学生信息进行增删改查操作。成绩信息管理： 管理员可以对学生成绩信息进行增删改查操作。志愿信息管理： 管理员可以查看和修改学生的志愿信息。学校专业管理：管理员可以对学校和专业进行增删改查

5. 其他功能权限管理： 系统应该对不同用户角色的权限进行管理，例如学生只能填报志愿和查看成绩，管理员拥有更多权限。数据统计： 系统应该能够对志愿情况、成绩等数据进行统计分析，以便于学校决策。

1. 业务流分析

业务流分析模块的主要目标是明确系统中各个功能模块之间的关系和操作流程。在这个模块中，首先绘制系统功能结构图，确定系统中所有子系统和功能模块，然后通过业务流程图展示各个功能模块之间的交互和操作步骤。这有助于理解系统在实际操作中的工作方式和流程。具体步骤包括：

* **系统功能结构图**：展示系统的主要功能和子功能模块，如用户管理、学生功能、管理员功能等。
* **业务流程图**：详细描述各个功能模块之间的操作流程，比如用户注册和登录、学生填报志愿、管理员审核志愿等。

通过业务流分析，可以清晰地看到系统各部分的功能和交互，有助于后续的系统开发和优化​​。

图1. 系统功能结构图

图2. 业务流程图

1. 数据流分析

数据流分析模块的目标是明确系统中数据的流动路径和处理过程。在这个模块中，首先绘制系统的顶层数据流图，接着分层次绘制详细的数据流图，展示数据在系统中从输入到输出的具体流动路径。主要步骤包括：

* **顶层数据流图**：概括性地展示系统中主要数据流动，如用户数据、学生数据、志愿数据等。
* **第1层数据流图**：详细描述每个主要数据流的具体流动和处理过程，比如用户注册、学生填报志愿、管理员审核等数据处理步骤。
* **第2层数据流图**：进一步细化第1层数据流图中的复杂数据流动，展示数据在各个子系统和模块中的流动路径和处理过程。

数据流分析帮助理解系统中的数据如何在不同模块之间流动和处理，确保数据在整个系统中的完整性和一致性​​。

图3.1顶层数据流图

图3.2第1层数据流图



图3.3 第2层数据流图

1. 数据字典
2. 数据项









1. 数据结构



1. 数据流





1. 数据库概念结构设计
2. 实体分析

该系统实体主要有用户、学生、管理员、班级、学校、专业、志愿

1. 属性分析

**实体属性：**

用户：登录帐号，密码，学生号，管理员号

学生：学生号，姓名，班级，志愿

管理员：管理员号，姓名

班级：班级号，学生号

学校：学校号，学校名

专业：专业号，专业名

志愿：学生号，学校号，专业号

**联系属性**

登录：帐号，密码

选择：志愿号

属于：班级号

管理：专业号，学校号

组成：学号，专业号，学校号



图4. 高考志愿辅助填报系统ER关系图

1. 联系分析

用户登录分为两种，其中一种是学生用户，另一种是管理员

一个学生属于一个班级，每个班级有多个学生

一个学生拥有一个高考成绩，一个成绩对应一个学生

一个学生可以选择多个志愿，每个志愿对应一个学生

每个志愿由一个学校和一个专业组成

一个管理员可以管理多个专业和学校信息

1. 概念模型分析（.CDM图）

通过powerDesigner创建该系统的概念模型设计图，如图：



图5. 高考志愿填报系统概念模型CDM图

1. 数据库逻辑结构设计
2. 概念模型转化为逻辑模型
3. 一对一关系的转化

用户（帐号，密码，学生号，管理员号）

学生（学生号，姓名，班级号，志愿号，成绩号）

管理员（管理员号，姓名）

成绩（成绩号，语文成绩，数学成绩，英语成绩，综合成绩）

1. 一对多关系的转化

学生（学生号，姓名，班级号，志愿号，成绩号）

班级（班级号，学生号）

1. 多对多关系的转化

学生（学生号，姓名，班级号，志愿号，成绩号）

学生志愿（志愿号，学生号，学校号，专业号，录取分数）

学校（学校号，学校名）

专业（专业号，专业名）

1. 逻辑模型设计（.LDM图）

通过powerDesigner将该系统的概念模型设计图转换为逻辑模型图，如图：



图6. 高考志愿辅助填报系统逻辑模型LDM图

1. 数据库物理结构设计
2. 表设计

　从概念性的建模过程中，我们了解到这个体系共有10个关联模式。每一种类型都与一种类型的类型相匹配，因此数据库系统共创建9张表。

表1 用户信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| log\_id  stu\_id  admin\_id  password  mode | varchar  bigint  bigint  varchar  int | 100  30 | 否  否  否  否  否 | 用户账号（主键自增）  学生号  管理员号  密码  用户模式 |

表2学生信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| stu\_id  log\_id  class\_id  score\_id  stu\_name | bigint  varchar  varchar  bigint  varchar | 100  30  30 | 否  否  否  否 | 学生号（主键自增）  用户账号  班级号  成绩号  学生姓名 |

表3管理员信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| admin\_id  log\_id  admin\_name | bigint  varchar  varchar | 100  30 | 否  否  否 | 管理员号（主键自增）  用户账号  管理员姓名 |

表4班级信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| class\_id | varchar | 30 | 否 | 班级号（主键自增） |

表5成绩表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| score\_id  stu\_id  chinese  math  english  overall | bigint  bigint  int  int  int  int |  | 否  否  否  否  否  否 | 成绩号（主键自增）  学生号  语文  数学  英语  综合 |

表6专业信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| major\_id  major\_name | bigint  varchar | 100 | 否  否 | 专业号（主键自增）  专业名 |

表7学校信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| school\_id  school\_name | bigint  varchar | 30 | 否  否 | 学校号（主键自增）  学校名 |

表8学生志愿信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| vul\_id  major\_id  stu\_id  school\_id | bigint  bigint  bigint  bigint |  | 否  否  否  否 | 志愿号（主键自增）  专业号  学生号  学校号 |

表9学校专业信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 类型 | 大小 | 是否为空 | 说明 |
| rel\_id  school\_id  major\_id  rel\_score | bigint  bigint  bigint  bigint |  | 否  否  否  否 | 关系号（主键自增）  学校号  专业号  专业分数 |

1. 创建表和完整性约束代码设计

create table Admin (

admin\_id varchar(30) not null,

log\_id varchar(100) not null,

admin\_name varchar(30) not null,

constraint PK\_ADMIN primary key nonclustered (admin\_id)

)

create table Class (

class\_id varchar(30) not null,

constraint PK\_CLASS primary key nonclustered (class\_id)

)

create table Grade (

score\_id bigint identity(1000,1) not for replication,

stu\_id bigint not null,

chinese int not null,

math int not null,

english int not null,

overall int not null,

constraint PK\_GRADE primary key nonclustered (score\_id)

)

create table Major (

major\_id bigint identity(1000,1) not for replication,

major\_name varchar(30) not null,

constraint PK\_MAJOR primary key nonclustered (major\_id)

)

create table Sch\_may\_rel (

rel\_id\_2 bigint identity(1000,1) not for replication,

school\_id bigint not null,

major\_id bigint not null,

rel\_sore int null,

constraint PK\_SCH\_MAY\_REL primary key nonclustered (rel\_id\_2)

)

create table School (

school\_id bigint identity(1000,1) not for replication,

school\_name varchar(30) not null,

constraint PK\_SCHOOL primary key nonclustered (school\_id)

)

create table Stu\_vul (

vul\_id bigint identity(1000,1) not for replication,

major\_id bigint null,

stu\_id bigint null,

school\_id bigint null,

constraint PK\_STU\_VUL primary key nonclustered (vul\_id)

)

create table Student (

stu\_id bigint identity(1000,1) not for replication,

stu\_name varchar(30) not null,

log\_id varchar(100) not null,

class\_id varchar(30) not null,

score\_id bigint null,

constraint PK\_STUDENT primary key nonclustered (stu\_id)

)

create table User\_info (

log\_id varchar(100) not null,

stu\_id bigint null,

admin\_id varchar(30) null,

password varchar(30) not null,

mode int not null,

constraint PK\_USER\_INFO primary key nonclustered (log\_id)

)

1. 创建物理模型设计图（.PDM）

通过Powerdesigner将逻辑模型设计中的.LDM图转化为物理模型设计图，如图



图7. 高考志愿辅助填报系统物理模型PDM图

1. 视图、索引、存储过程和触发器

索引创建

create index Relationship\_2\_FK on Admin (

log\_id ASC

)

create index Relationship\_3\_FK on Grade (

stu\_id ASC

)

create index Relationship\_6\_FK on Sch\_may\_rel (

school\_id ASC

)

create index Relationship\_7\_FK on Sch\_may\_rel (

major\_id ASC

)

create index Relationship\_9\_FK on Stu\_vul (

stu\_id ASC

)

create index Relationship\_10\_FK on Stu\_vul (

major\_id ASC

)

create index Relationship\_11\_FK on Stu\_vul (

school\_id ASC

)

create index Relationship\_1\_FK on Student (

log\_id ASC

)

create index Relationship\_3\_FK on Student (

score\_id ASC

)

create index Relationship\_8\_FK on Student (

class\_id ASC

)

create index Relationship\_1\_FK on User\_info (

stu\_id ASC

)

create index Relationship\_2\_FK on User\_info (

admin\_id ASC

)

存储过程

select \* from Stu\_vul;

-- 创建一个登录过程

create procedure Login

@sno varchar(100),@pw varchar(30)

as

select COUNT(\*) as cnt from User\_info

where log\_id=@sno and password=@pw;

-- 查找模式

create procedure Find\_mode

@sno char(100)

as

select mode from User\_info

where log\_id = @sno;

-- 创建一个学生系统信息过程

create procedure Stud

@sno varchar(100)

as

select \* from Student

where log\_id=@sno;

-- 创建一个查询学生成绩过程

create procedure Scla

@sno bigint

as

select chinese, math, english, overall, (chinese + math + english + overall) as sum

from Grade

where stu\_id = @sno;

-- 创建一个查询各个学校录取最低分

CREATE PROCEDURE Find\_sch

AS

BEGIN

SELECT

s.school\_id as sch\_id,

s.school\_name AS sch\_name,

MIN(sr.rel\_sore) AS min\_score

FROM

School s

JOIN

Sch\_may\_rel sr ON s.school\_id = sr.school\_id

JOIN

Major m ON sr.major\_id = m.major\_id

GROUP BY

s.school\_name, s.school\_id;

END;

-- 创建一个查询学校号

create procedure Find\_sch\_id

@m\_sch\_name varchar(100)

as

select School.school\_id as id

from School

where School.school\_name = @m\_sch\_name;

-- 创建一个查询学校专业

create procedure Find\_sch\_may

@m\_sch\_id int

as

select Major.major\_id as id, Major.major\_name as major\_name, Sch\_may\_rel.rel\_sore as rel\_soce

from Major, Sch\_may\_rel

where Sch\_may\_rel.school\_id = @m\_sch\_id and Sch\_may\_rel.major\_id = Major.major\_id;

-- 创建一个查询学生志愿

create procedure Find\_stu\_vul

@m\_id bigint

as

select distinct School.school\_name as sch, Major.major\_name as may

from Student, School, Major, Stu\_vul

where Stu\_vul.stu\_id = @m\_id and Stu\_vul.school\_id = School.school\_id and Stu\_vul.major\_id = Major.major\_id;

-- 创建一个检查学校号是否存在

create procedure check\_sch

@stu\_id int

as

select COUNT(\*) as cnt from School

where School.school\_id = @stu\_id;

-- 创建一个检查专业号是否存在

create procedure check\_may

@may\_id int

as

select COUNT(\*) as cnt from Major

where Major.major\_id = @may\_id;

-- 创建一个检查志愿是否重复

create procedure check\_dup

@stu\_id int, @sch\_id int, @may\_id int

as

select count(\*) as cnt from Stu\_vul

where Stu\_vul.stu\_id = @stu\_id and Stu\_vul.school\_id = @sch\_id and Stu\_vul.major\_id = @may\_id;

-- 创建一个插入学生志愿

create procedure add\_stu\_vul

@m\_stu\_id int, @m\_sch\_id int, @m\_may\_id int

as

insert into Stu\_vul(stu\_id, school\_id, major\_id)

values (@m\_stu\_id, @m\_sch\_id, @m\_may\_id);

-- 创建一个删除学生志愿

create procedure del\_stu\_vul

@m\_stu\_id int, @m\_sch\_id int, @m\_may\_id int

as

delete from Stu\_vul

where @m\_stu\_id = Stu\_vul.stu\_id and @m\_sch\_id = Stu\_vul.school\_id and @m\_may\_id = Stu\_vul.major\_id;

-- 创建一个查询学校和专业

create procedure Find\_admin\_sch\_may

as

select distinct sm.school\_id as sch\_id, s.school\_name as sch\_name, sm.major\_id as m\_id, m.major\_name as m\_name, sm.rel\_sore as score

from Sch\_may\_rel sm, School s, Major m

where sm.school\_id = s.school\_id and sm.major\_id = m.major\_id;

-- 查询班级成绩最大值最小值平均值

CREATE PROCEDURE Find\_score

@m\_class INT,

@m\_cou VARCHAR(100)

AS

BEGIN

DECLARE @sql NVARCHAR(MAX);

-- Construct the dynamic SQL query

SET @sql = N'

SELECT

MAX(g.' + QUOTENAME(@m\_cou) + N') AS Highest\_Score,

MIN(g.' + QUOTENAME(@m\_cou) + N') AS Lowest\_Score,

AVG(g.' + QUOTENAME(@m\_cou) + N') AS Average\_Score

FROM

Student s

JOIN

Grade g ON s.score\_id = g.score\_id

WHERE

s.class\_id = @m\_class;

';

-- Execute the dynamic SQL query

EXEC sp\_executesql @sql, N'@m\_class INT', @m\_class;

END;

1. 数据库功能实现及界面展示

本系统基于MFC、sqlSever、powerDesinger实现大致分为三个模块，模块中实现了基本数据库增删改查操作，并且系统界面简洁实用。

1. 登录模块

用户可以在登录时输入帐号和密码，每个帐号密码对应一个用户，此系统共有两种用户权限。



图8.

如果输入错误会进行提示



图9.

1. 学生模块

登录学生帐号后的界面，大致有五个功能，同时也将学生信息显示出来



图10.

查看成绩界面

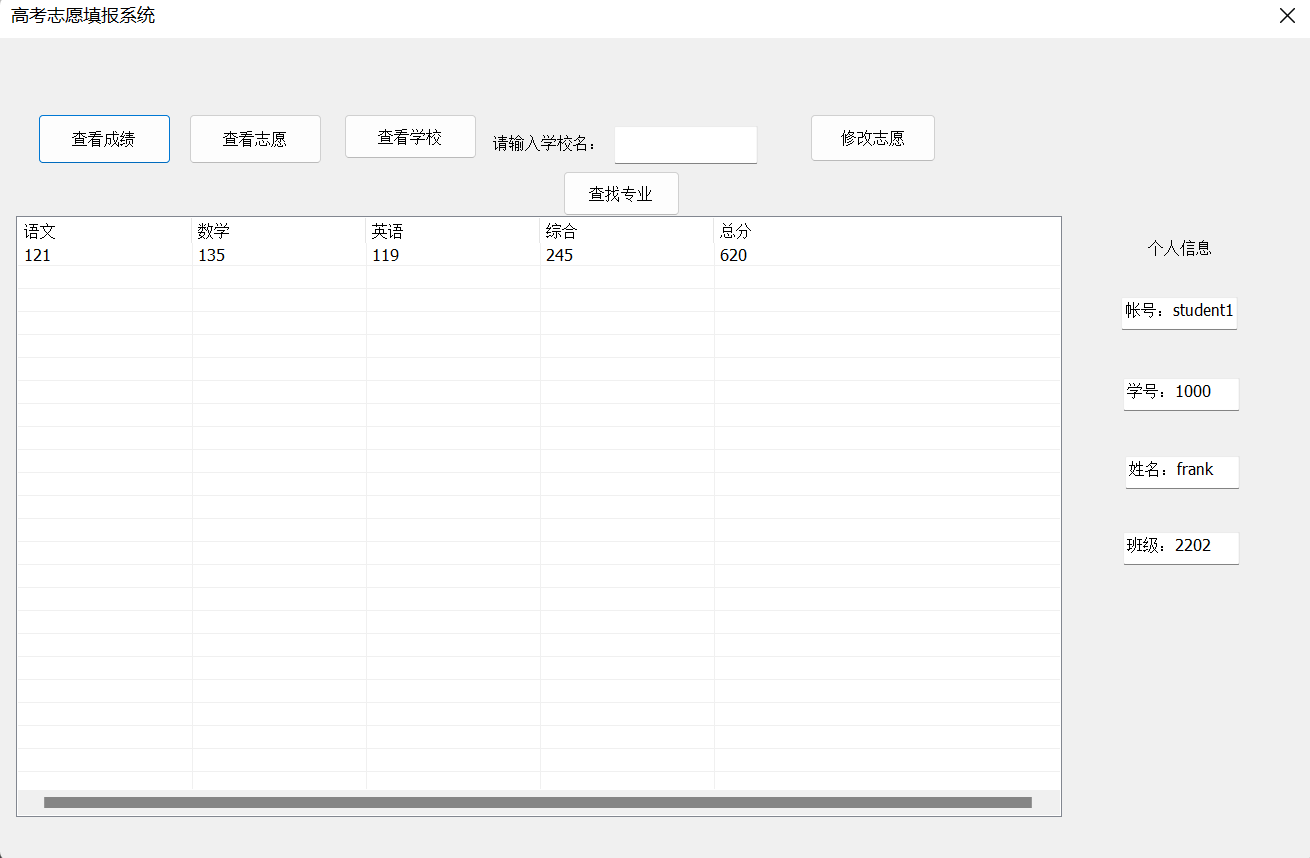


图11

查看志愿界面

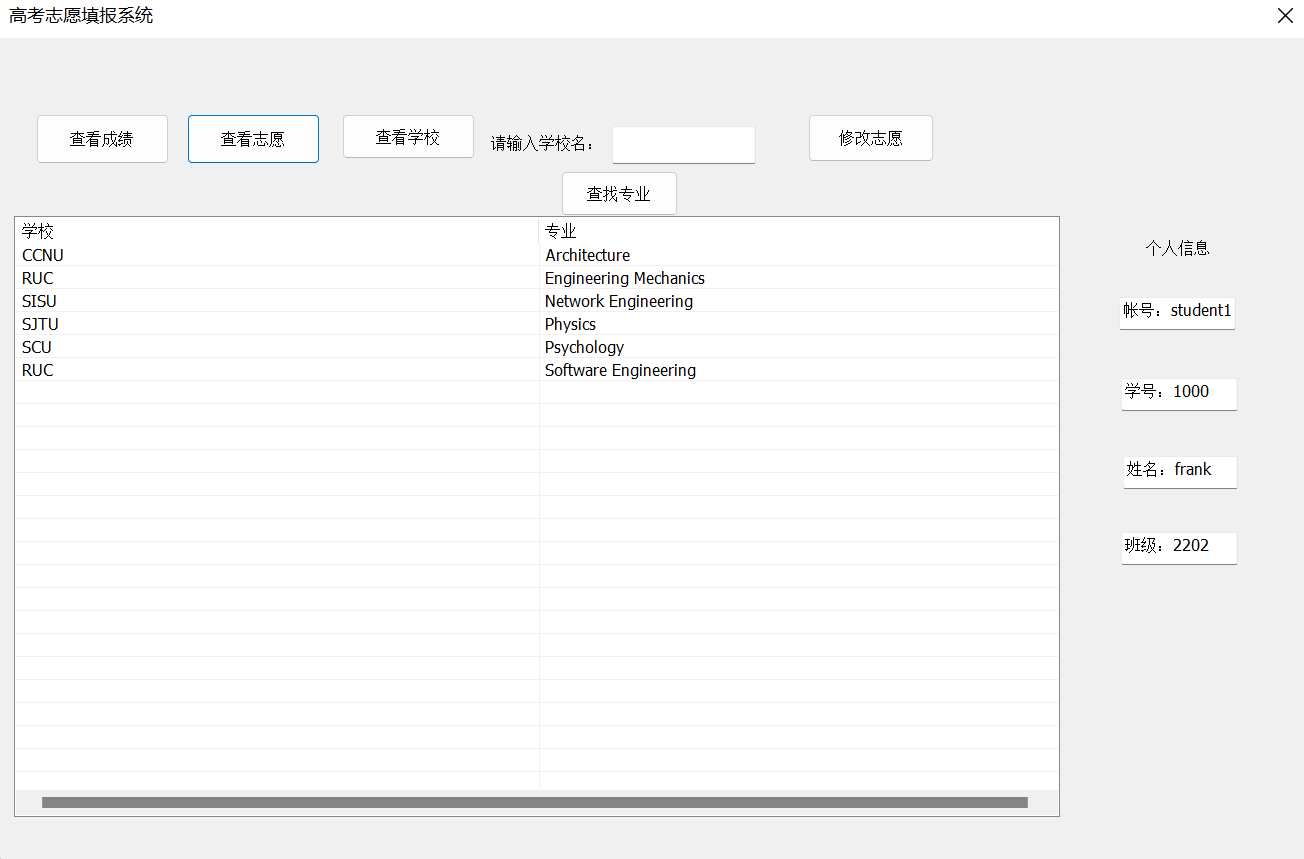


图12

查看学校界面



图13

输入正确学校名后查看专业

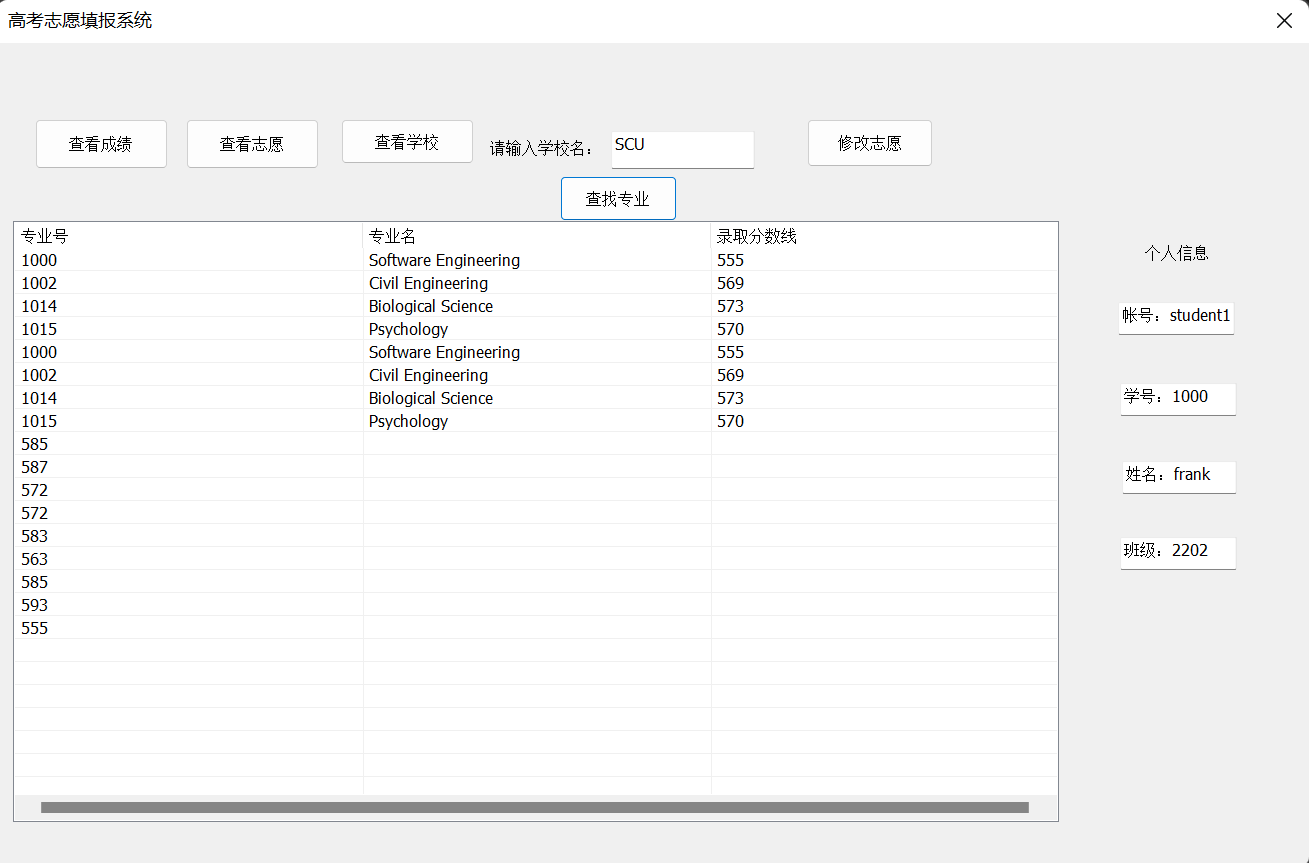


图14

修改志愿(包含添加和删除学生填写的志愿)

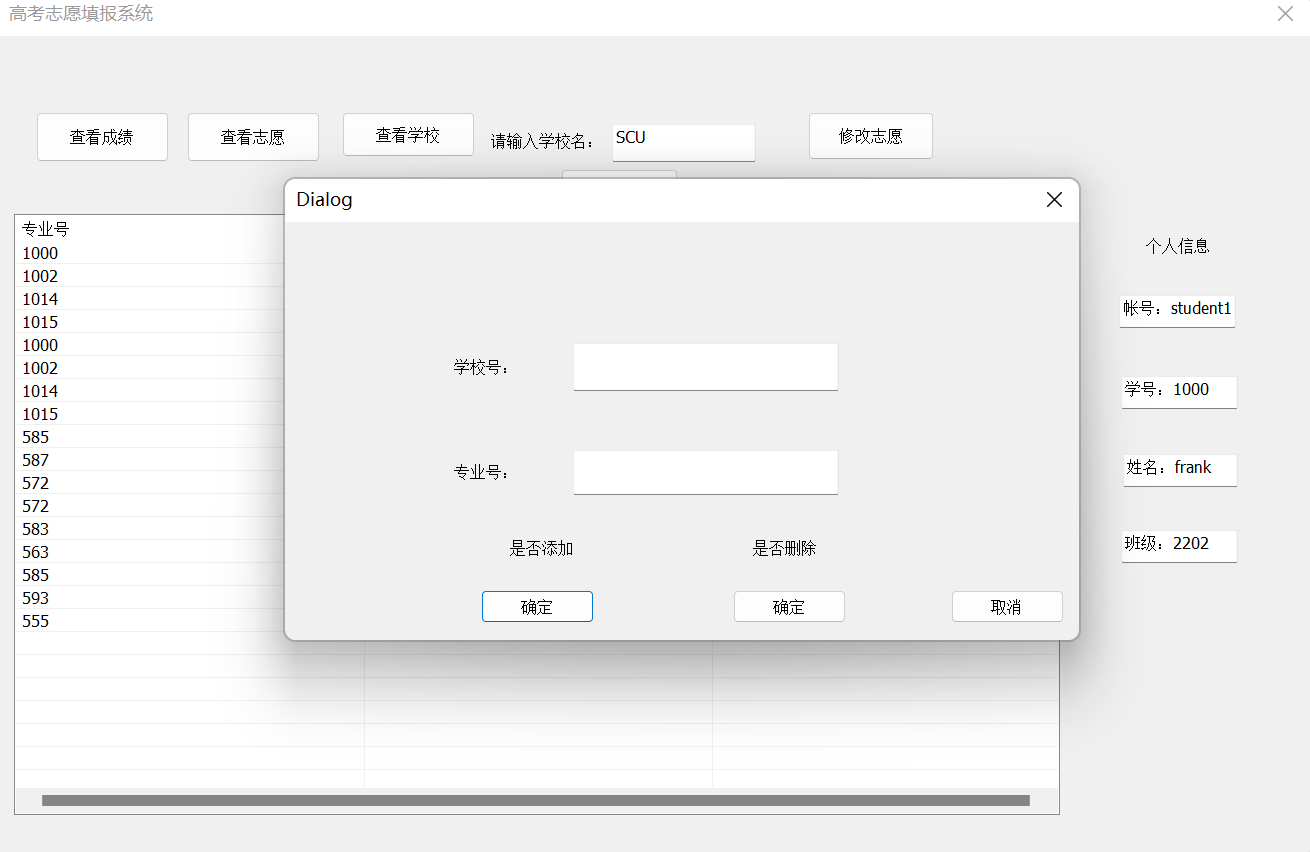


图15

输入不存在的学校号或者专业号会提示

1. 管理员模块

登录管理员帐号后的界面，大致有三个功能区

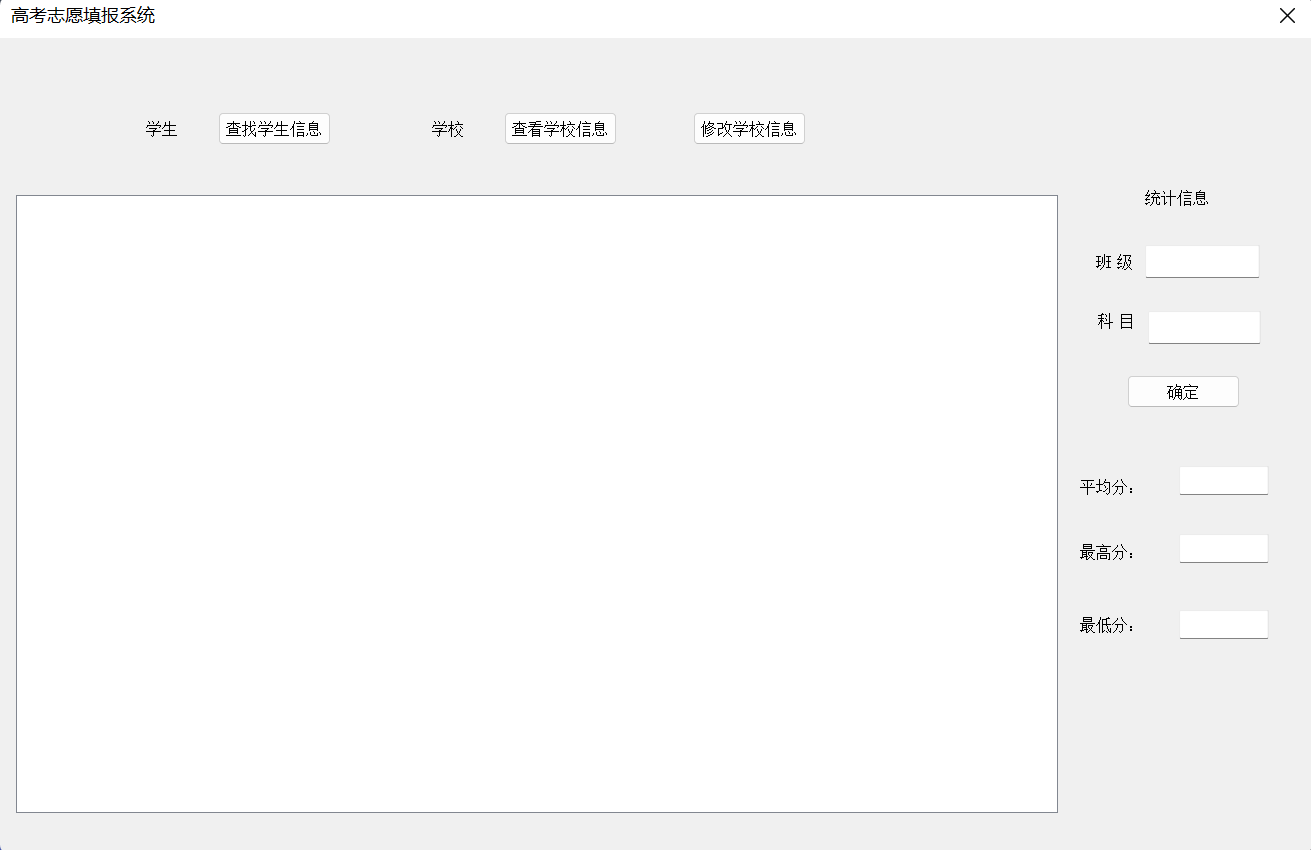


图16.

查找学生信息

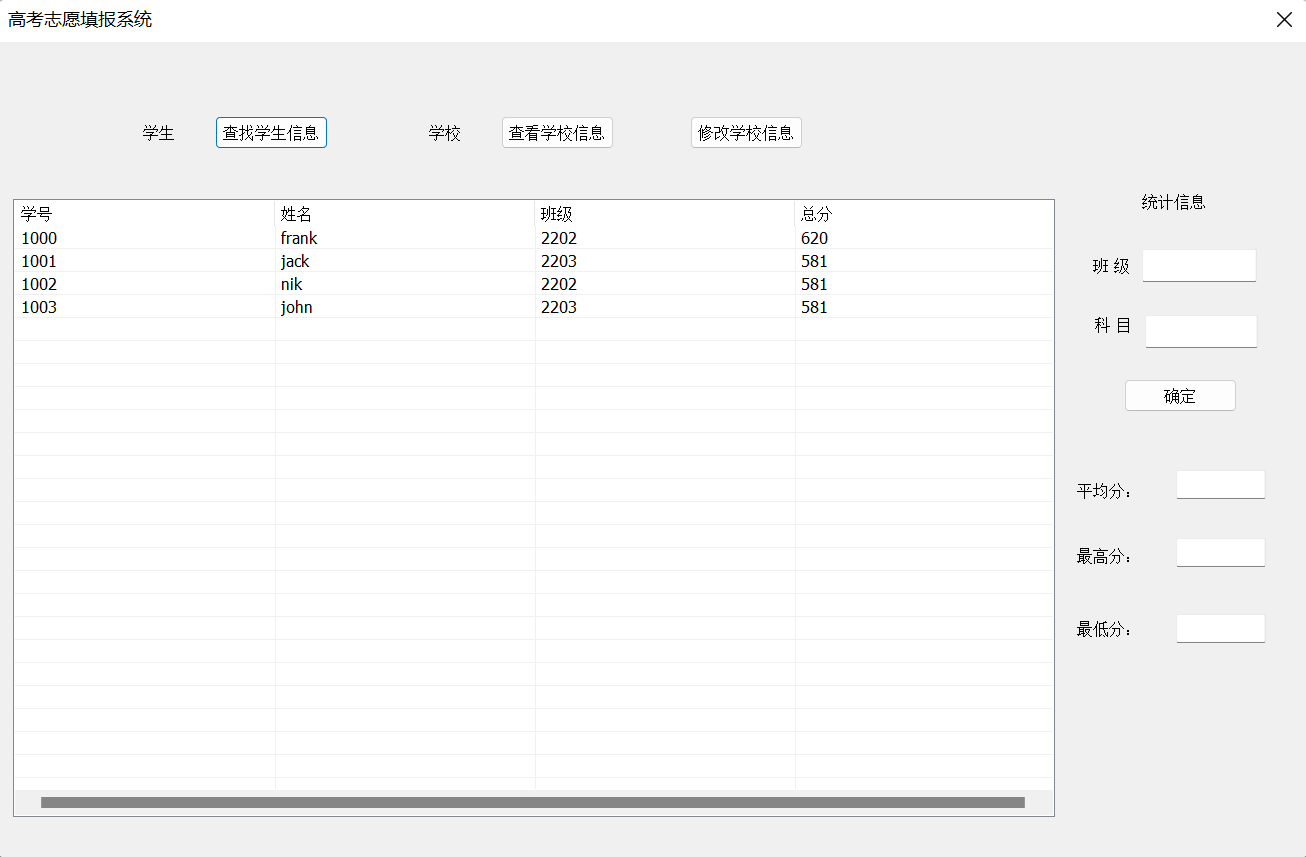


图17.

查看学校信息(将所有学校所包含的专业显示)

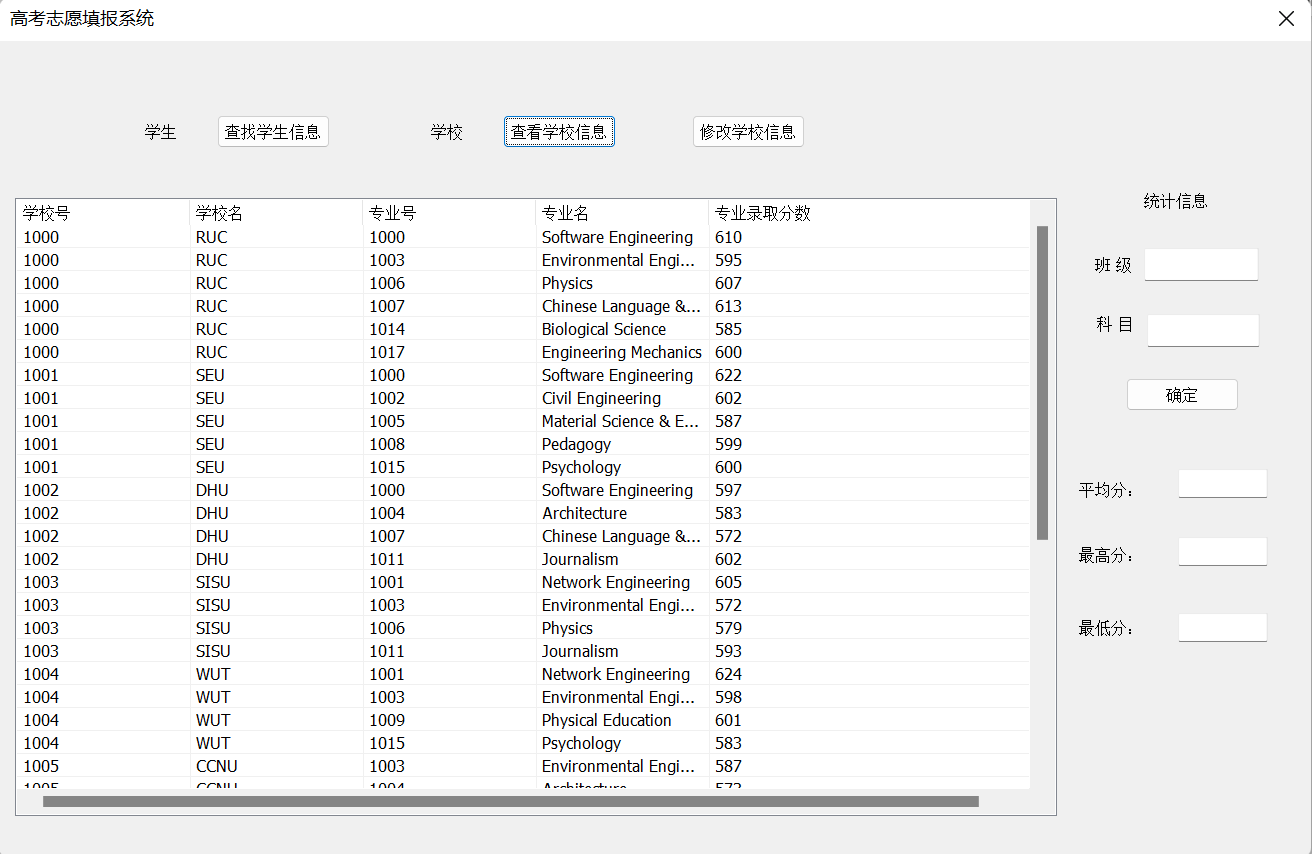


图18.

修改学校信息（包含添加学校和专业以及录取分数）

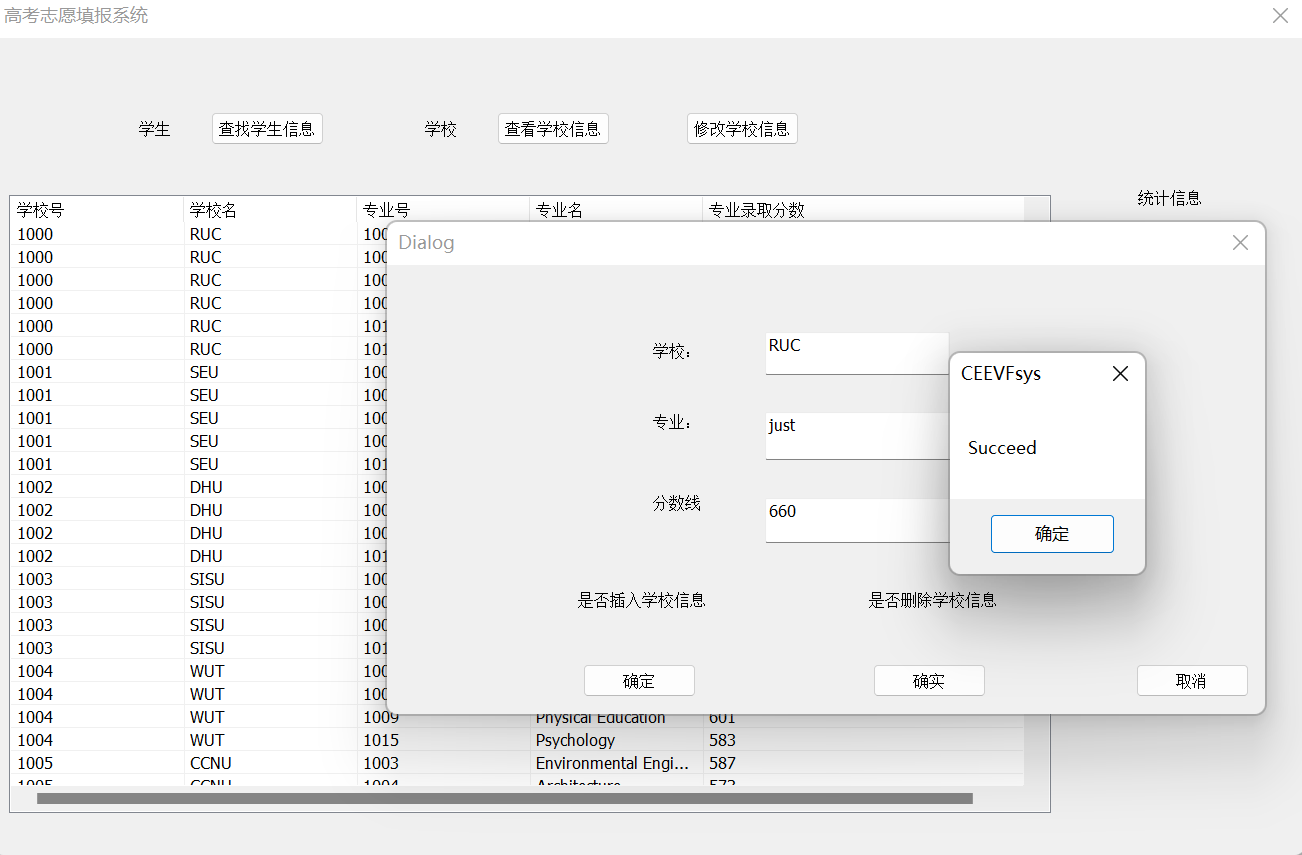


图19.

查看统计信息（输入班级和科目查看最高最低分以及平均分）

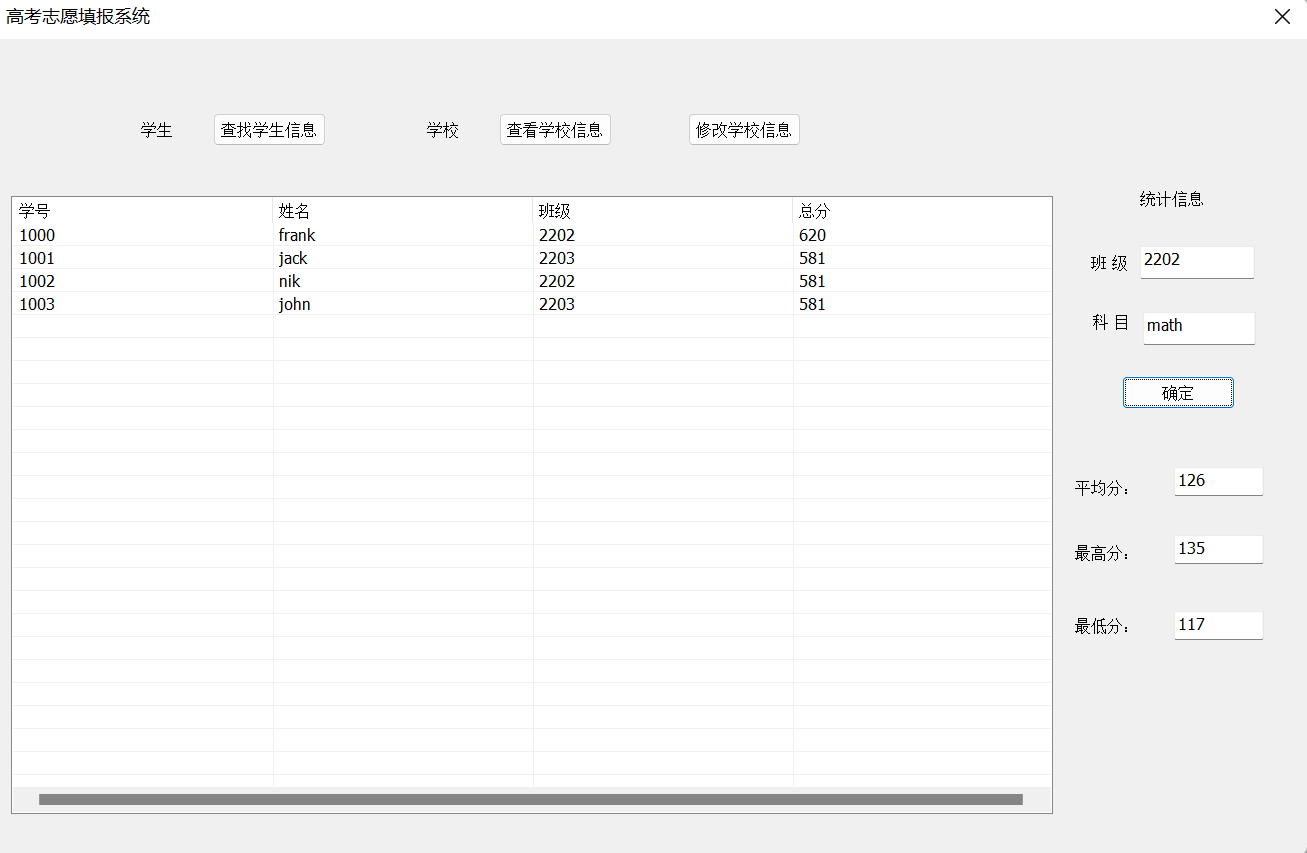


图20.

1. 总结与展望

总结

在进行数据库系统的设计时，最重要的是要对其所要实现的功能进行分析，并要对各种软件进行正确的应用， 其中最大的问题就是在关系模型的设计和系统接口的设计中，所面临的问题和解决方法是：

1.由于 PowerDesigner系统对实体之间连接关系的研究不足，使得 PowerDesigner系统中的内外关键字的设计存在 困难，使得系统中的整体结构不能被直接提取出来。最终又进行了再分析。

2.在进行创建MFC对话框时由于不熟悉各种工具的使用，使得在项目初期设计窗口十分艰难，最终通过不断的上网搜索相关教程，结合项目本身不断调试，最终能够实现一个功能简单的系统。

3.在设计不同窗口代码中需要引入类似的用于连接数据库的代码，用于将MFC应用配合sqlSever中创建的存储过程实现不同功能的连接。

4.在设计listControl时由于粗心导致窗口中的字段数与查询的结果字段数不匹配，从而发生程序异常。

5.每当引入一个控件时都要记住其属性名，否在会在其他函数体当中调用时混乱。

6.在一个控件调用另一个窗口时，需要做好连接工作，使其能够在子窗口中正确执行代码并与数据库完成交互。

展望

　经过这次的资料库课程的开发，本人深切地感受到，要想完成一套完善的资料库的开发，就必须要有一种认真 的工作态度与耐性。在进行设计的时候，要参考很多的材料，向别人学习并吸取别人的经验与教训，多与别人一起 探讨，可以帮助我们更好地消化所学。在这个过程中，我也遇到了很多问题，大多数都已经得到了解决，只有一小 部分还没有得到解决。但是，我坚信，只要我不断地努力，我就能从老师那里得到更多的信息，得到更多的信息。 我将在今后的工作中，努力提高自己的判断力，并努力提高自己的判断力。